# strongSwan - Bug #2815

## IPSec SAs disappear after some time

02.11.2018 08:49 - Sony Arpita Das

| | | | |
|---|---|---|---|
| **Status:** | Closed | **Start date:** | |
| **Priority:** | Normal | **Due date:** | |
| **Assignee:** | Tobias Brunner | **Estimated time:** | 0.00 hour |
| **Category:** | libcharon | | |
| **Target version:** | 5.7.2 | | |
| **Affected version:** | 5.6.3 | **Resolution:** | Fixed |

**Description**

Hi,

Following is my configuration on 2 machines -

Host1

```
cat swanctl.conf
connections {
  gw-gw {
     local_addrs  = 102.1.1.21
     remote_addrs = 102.1.1.96
     local {
        auth = pubkey
        certs = cool1Cert.der
        id = "C=CH, O=blr.asicdesigners.com, CN=cool1"
     }
     remote {
        auth = pubkey
        id = "C=CH, O=blr.asicdesigners.com, CN=viper1"
     }
     children {
        net-net {
           rekey_time = 2000
           rekey_bytes = 500000000
           rekey_packets = 1000000
           esp_proposals = aes128gcm96-x25519-esn-noesn
           hw_offload = no
        }
     }
     version = 2
     mobike = no
     reauth_time = 1080
     proposals = aes128-sha1-x25519
  }
}
```

Host2

```
cat swanctl.conf
connections {
  gw-gw {
     local_addrs  = 102.1.1.96
     remote_addrs = 102.1.1.21
     local {
        auth = pubkey
        certs = viper1Cert.der
        id = "C=CH, O=blr.asicdesigners.com, CN=viper1"
     }
     remote {
```

```
        auth = pubkey
        id = "C=CH, O=blr.asicdesigners.com, CN=cool1"
     }
     children {
        net-net {
           rekey_time = 2000
           rekey_bytes = 500000000
           rekey_packets = 1000000
           esp_proposals = aes128gcm96-x25519-esn-noesn
           hw_offload = no
        }
     }
     version = 2
     mobike = no
     reauth_time = 1080
     proposals = aes128-sha1-x25519
  }
}
```

After about an hour, the SAs disappear and IPSec tunnels die and never come back on.

### Associated revisions

**Revision ecba84a0 - 22.11.2018 11:31 - Tobias Brunner**

child-delete: Don't send delete for expired CHILD_SAs that were already rekeyed

The peer might not have seen the CREATE_CHILD_SA response yet, receiving a DELETE for the SA could then trigger it to abort the rekeying, causing the deletion of the newly established SA (it can't know whether the DELETE was sent due to an expire or because the user manually deleted it). We just treat this SA as if we received a DELETE for it. This is not an ideal situation anyway, as it causes some traffic to get dropped, so it should usually be avoided by setting appropriate soft and hard limits.

References #2815.

### History

**#1 - 05.11.2018 10:11 - Tobias Brunner**

*- Description updated*

*- Status changed from New to Feedback*

> After about an hour, the SAs disappear and IPSec tunnels die and never come back on.

So read the logs, they should tell you what happens and why the SAs are closed. Please also read ExpiryRekey.

**#2 - 06.11.2018 10:03 - Sony Arpita Das**

*- File logs.txt added*

Hi Tobias,

I have attached the complete logs and the SA's close with the following message -

```
Nov  5 23:52:19 cool1 charon: 07[KNL] creating rekey job for CHILD_SA ESP/0xc24f3e08/102.1.1.96
Nov  5 23:52:19 cool1 charon: 07[IKE] establishing CHILD_SA net-net{108} reqid 5
Nov  5 23:52:19 cool1 charon: 07[ENC] generating CREATE_CHILD_SA request 126 [ N(REKEY_SA) SA No KE TSi TSr ]
Nov  5 23:52:19 cool1 charon: 07[NET] sending packet: from 102.1.1.21[500] to 102.1.1.96[500] (252 bytes)
Nov  5 23:52:19 cool1 charon: 10[NET] received packet: from 102.1.1.96[500] to 102.1.1.21[500] (236 bytes)
Nov  5 23:52:19 cool1 charon: 10[ENC] parsed CREATE_CHILD_SA response 126 [ SA No KE TSi TSr ]
Nov  5 23:52:19 cool1 charon: 10[IKE] inbound CHILD_SA net-net{108} established with SPIs cde1c53f_i cf5f85a8_
o and TS 102.1.1.21/32 === 102.1.1.96/32
Nov  5 23:52:19 cool1 charon: 10[IKE] outbound CHILD_SA net-net{108} established with SPIs cde1c53f_i cf5f85a8
_o and TS 102.1.1.21/32 === 102.1.1.96/32
Nov  5 23:52:19 cool1 charon: 10[IKE] closing CHILD_SA net-net{107} with SPIs c7c34f04_i (202173 bytes) c24f3e
08_o (467166474 bytes) and TS 102.1.1.21/32 === 102.1.1.96/32
Nov  5 23:52:19 cool1 charon: 10[IKE] sending DELETE for ESP CHILD_SA with SPI c7c34f04
```

```
Nov  5 23:52:19 cool1 charon: 10[ENC] generating INFORMATIONAL request 127 [ D ]
Nov  5 23:52:19 cool1 charon: 10[NET] sending packet: from 102.1.1.21[500] to 102.1.1.96[500] (76 bytes)
Nov  5 23:52:19 cool1 charon: 09[NET] received packet: from 102.1.1.96[500] to 102.1.1.21[500] (76 bytes)
Nov  5 23:52:19 cool1 charon: 09[ENC] parsed INFORMATIONAL response 127 [ D ]
Nov  5 23:52:19 cool1 charon: 09[IKE] received DELETE for ESP CHILD_SA with SPI c24f3e08
Nov  5 23:52:19 cool1 charon: 09[IKE] CHILD_SA closed
Nov  5 23:52:20 cool1 charon: 04[KNL] creating rekey job for CHILD_SA ESP/0xcf5f85a8/102.1.1.96
Nov  5 23:52:20 cool1 charon: 04[IKE] establishing CHILD_SA net-net{109} reqid 5
Nov  5 23:52:20 cool1 charon: 04[ENC] generating CREATE_CHILD_SA request 128 [ N(REKEY_SA) SA No KE TSi TSr ]
Nov  5 23:52:20 cool1 charon: 04[NET] sending packet: from 102.1.1.21[500] to 102.1.1.96[500] (252 bytes)
Nov  5 23:52:20 cool1 charon: 05[KNL] creating delete job for CHILD_SA ESP/0xcf5f85a8/102.1.1.96
Nov  5 23:52:20 cool1 charon: 14[KNL] creating acquire job for policy 102.1.1.21/32[udp/48273] === 102.1.1.96/
32[udp/33247] with reqid {5}
Nov  5 23:52:20 cool1 charon: 14[CFG] trap not found, unable to acquire reqid 5
Nov  5 23:52:24 cool1 charon: 08[IKE] retransmit 1 of request with message ID 128
Nov  5 23:52:24 cool1 charon: 08[NET] sending packet: from 102.1.1.21[500] to 102.1.1.96[500] (252 bytes)
Nov  5 23:52:31 cool1 charon: 11[IKE] retransmit 2 of request with message ID 128
Nov  5 23:52:31 cool1 charon: 11[NET] sending packet: from 102.1.1.21[500] to 102.1.1.96[500] (252 bytes)
Nov  5 23:52:36 cool1 charon: 07[NET] received packet: from 102.1.1.96[500] to 102.1.1.21[500] (76 bytes)
Nov  5 23:52:36 cool1 charon: 07[ENC] parsed INFORMATIONAL request 85 [ D ]
Nov  5 23:52:36 cool1 charon: 07[IKE] received DELETE for ESP CHILD_SA with SPI cf5f85a8
Nov  5 23:52:36 cool1 charon: 07[IKE] closing CHILD_SA net-net{108} with SPIs cde1c53f_i (310550 bytes) cf5f85
a8_o (0 bytes) and TS 102.1.1.21/32 === 102.1.1.96/32
Nov  5 23:52:36 cool1 charon: 07[IKE] sending DELETE for ESP CHILD_SA with SPI cde1c53f
Nov  5 23:52:36 cool1 charon: 07[IKE] CHILD_SA closed
Nov  5 23:52:36 cool1 charon: 07[IKE] detected CHILD_REKEY collision with CHILD_DELETE
```

Please give us a hint as to what is causing the issue.

Thanks,
Sony

#### #3 - 06.11.2018 11:17 - Tobias Brunner

*- Affected version changed from 5.7.1 to 5.6.3*

First, you might want to reconsider using *reauth_time* (refer to the page I linked before), or at least consider using make-before-break reauth.

Then you might want to consider increasing *rekey_bytes* (or *life_bytes*), because the connection seems to be quite busy:

```
Nov  5 23:43:06 cool1 charon: 09[IKE] CHILD_SA net-net{5} established with SPIs c7378e95_i cc3f4187_o and TS 1
02.1.1.21/32 === 102.1.1.96/32
...
Nov  5 23:47:02 cool1 charon: 12[KNL] creating rekey job for CHILD_SA ESP/0xcc3f4187/102.1.1.96
```

So these ~500000000 bytes (476 MiB) were transmitted in less than a minute. Actually, for the next SA it was even quicker:

```
Nov  5 23:47:02 cool1 charon: 14[IKE] outbound CHILD_SA net-net{6} established with SPIs c66f6e1f_i cb11ebee_o
 and TS 102.1.1.21/32 === 102.1.1.96/32
...
Nov  5 23:47:05 cool1 charon: 08[KNL] creating rekey job for CHILD_SA ESP/0xcb11ebee/102.1.1.96
```

Since the default value for *life_bytes* is 10% higher than *rekey_bytes* and *rand_bytes* lowers the potential rekey point by the same amount there might not be much time to actually do the rekeying (between 10% and 20% of *rekey_bytes*). In fact, here a delete for the SA is triggered before the rekeing was complete:

```
Nov  5 23:47:06 cool1 charon: 04[KNL] creating delete job for CHILD_SA ESP/0xcb11ebee/102.1.1.96
...
Nov  5 23:47:09 cool1 charon: 14[IKE] outbound CHILD_SA net-net{7} established with SPIs ca2285cc_i cb1c02f0_o
 and TS 102.1.1.21/32 === 102.1.1.96/32
```

That means there won't be an SA during that time (the kernel already removed the SA once the hard limit for transmitted bytes was reached).

All this is then repeated constantly (presumably on both sides, the log from there might help to see what exactly is going on there) causing exchange collisions and eventually this host ends up without a CHILD_SA somehow (the peer seems to delete the SA the host tries to rekey). In general, collisions should be handled correctly, but there might be an issue here that has not yet been recognized. One possibility is that the rekey and expire events from the kernel were triggered at about the same time on the peer, and that the job that deletes the CHILD_SA was executed by the thread pool before the job that would rekey it (as mentioned, the logs of the other peer of this same session might help, in particular at the end).

#### #4 - 06.11.2018 11:27 - Sony Arpita Das

*- File logs_from_peer.txt added*

Hi Tobias,

Thanks for the response, I will take into consideration all your inputs. In the meantime attached is the log from the peer

Thanks,
Sony

#### #5 - 06.11.2018 13:41 - Tobias Brunner

> In the meantime attached is the log from the peer

Thanks. It's similar to what I assumed:

```
Nov  5 23:52:24 viper1 charon: 13[KNL] creating rekey job for CHILD_SA ESP/0xcde1c53f/102.1.1.21
Nov  5 23:52:25 viper1 charon: 16[KNL] creating delete job for CHILD_SA ESP/0xcde1c53f/102.1.1.21
```

The two jobs are queued to the job queue about concurrently and while the rekey job probably runs first, it doesn't do anything because the CHILD_SA is already in state CHILD_REKEYED due to a handled request to rekey the CHILD_SA that was received earlier:

```
Nov  5 23:52:20 viper1 charon: 14[ENC] parsed CREATE_CHILD_SA request 128 [ N(REKEY_SA) SA No KE TSi TSr ]
Nov  5 23:52:20 viper1 charon: 14[IKE] inbound CHILD_SA net-net{114} established with SPIs c5f81b8d_i c0645012
_o and TS 102.1.1.96/32 === 102.1.1.21/32
Nov  5 23:52:20 viper1 charon: 14[ENC] generating CREATE_CHILD_SA response 128 [ SA No KE TSi TSr ]
```

The problem is that the delete job causes the host to properly delete the CHILD_SA:

```
Nov  5 23:52:25 viper1 charon: 16[IKE] closing expired CHILD_SA net-net{113} with SPIs cf5f85a8_i cde1c53f_o a
nd TS 102.1.1.96/32 === 102.1.1.21/32
Nov  5 23:52:25 viper1 charon: 16[IKE] sending DELETE for ESP CHILD_SA with SPI cf5f85a8
```

Unfortunately, the DELETE reaches the first host before the CREATE_CHILD_SA response (there are a bunch of retransmits seen in the log) and the collision handling there causes the rekeying to be aborted.

I think we can avoid this issue by not sending a delete for expired CHILD_SAs that were already rekeyed (they are expected to be deleted by the peer that initiated the rekeying anyway). I pushed a commit that does that to the *2815-delete-expired* branch. Handling the DELETE differently might also be something to consider, as the peer apparently handled the CREATE_CHILD_SA request properly (i.e. it didn't respond with an error notify because the old CHILD_SA was not found anymore or is currently being deleted), which gives us reason to assume that the DELETE was actually sent after the CREATE_CHILD_SA was already handled. However, that could also be the case when handling rekey collisions, i.e. if both initiate a rekeying concurrently and redundant SAs have to be deleted afterwards and some messages get delayed.

Anyway, you should definitely adjust the rekey settings so this situation does not occur in the first place (it causes unavoidable traffic loss because the SA is automatically removed from the kernel without a replacement being installed yet).

#### #6 - 06.11.2018 13:41 - Tobias Brunner

*- Tracker changed from Issue to Bug*

*- Assignee set to Tobias Brunner*

*- Target version set to 5.7.2*

#### #7 - 08.11.2018 12:01 - Sony Arpita Das

Hi Tobias,

I increased the "rekey_time", "reauth_time" and added "make_before_break = yes". Yet the SA breaks after few minutes. Following is the content of swanctl.conf and strongswan.conf -

onnections {
gw-gw {
local_addrs = 102.1.1.21
remote_addrs = 102.1.1.96
local {
auth = pubkey
certs = cool1Cert.der
id = "C=CH, O=blr.asicdesigners.com, CN=cool1"
}
remote {
auth = pubkey
id = "C=CH, O=blr.asicdesigners.com, CN=viper1"
}
children {
net-net {
rekey_time = 5400
rekey_bytes = 500000000

```
rekey_packets = 1000000
esp_proposals = aes128gcm96-x25519-esn-noesn
hw_offload = no
}
}
version = 2
mobike = no
reauth_time = 10800
proposals = aes128-sha1-x25519
}
}

[root@cool1 swanctl]# cat ../strongswan.conf

1. strongswan.conf - strongSwan configuration file #
2. Refer to the strongswan.conf(5) manpage for details #
3. Configuration changes should be made in the included files

charon {
load_modular = yes
plugins {
include strongswan.d/charon/*.conf
}
make_before_break = yes
}

include strongswan.d/*.conf
```

Thanks,
Sony

#### #8 - 08.11.2018 13:00 - Tobias Brunner

> I increased the "rekey_time", "reauth_time" and added "make_before_break = yes". Yet the SA breaks after few minutes.

Why would you think that changes anything? Please re-read my comments above, the problematic settings are your traffic based limits (*rekey_bytes* etc.).

#### #9 - 12.11.2018 07:35 - Sony Arpita Das

Hi Tobias,

We are evaluating 25G adapters and hence massive amount of data is pushed over the tunnel.
Is there anyway to set rekey_bytes as an infinite value ??

Thanks,
Sony

#### #10 - 12.11.2018 09:04 - Tobias Brunner

> Is there anyway to set rekey_bytes as an infinite value ??

If you don't want to rekey based on traffic, just don't set it at all. Again, please read ExpiryRekey.

#### #11 - 12.11.2018 09:29 - Sony Arpita Das

I also changed the rekey_bytes value to following, yet it didnt help

```
children {
net-net {
rekey_time = 5400
rekey_bytes = 50000000000000000
rekey_packets = 1000000
esp_proposals = aes128gcm96-x25519-esn-noesn
hw_offload = no
}
}
```

#### #12 - 14.11.2018 11:05 - Sony Arpita Das

Hi Tobias,

Thanks for the explanation.
Upon removing rekey_bytes from the configuration, the tunnel does not expire.

Thanks,
Sony

#### #13 - 14.11.2018 12:11 - Tobias Brunner

> Upon removing rekey_bytes from the configuration, the tunnel does not expire.

OK.

> I also changed the rekey_bytes value to following, yet it didnt help

How so? While 50000000000000000 is actually shortened to 500000000000000 (a 16 character buffer is currently used) this is still 454 TiB. So you should not reach the soft or hard limits in a long time (at the earliest a rekeying is initiated after 450000000000000 bytes = 409 TiB).

#### #14 - 22.11.2018 11:34 - Tobias Brunner

*- Category set to libcharon*

*- Status changed from Feedback to Closed*

*- Resolution set to Fixed*

### Files

| | | | |
|---|---|---|---|
| logs.txt | 373 KB | 06.11.2018 | Sony Arpita Das |
| logs_from_peer.txt | 379 KB | 06.11.2018 | Sony Arpita Das |